Departamento de Computación
FCEFQyN, Universidad Nacional de Río Cuarto
Asignatura: Estructuras de Datos y Algoritmos - Algoritmos I
Segundo Cuatrimestre de 2016

Práctica No. 2

Ej. 1. Lea las secciones 3.1 a la 3.5 del capítulo 3 de *Data Structures and Problem Solving Using Java* (Mark Allen Weiss 4th Edition).

Ej. 2. Bajar los archivos `ej1a.zip` y `ej1b.zip` de Moodle. Estos incluyen clases que modelan libros (`Libro.java`), y catálogos (conjuntos) de libros (`Catalogo.java`).

a) Compilar las clases `Libro.java` y `Main.java` contenidas en `ej1a.zip`. Ejecutar `Main`. Crear algunos (objetos) libros adicionales en Main.java, y mostarlos por pantalla.

b) Descomprimir `ej1b.zip`. Implementar el método `buscarPorTitulo` de la clase `Catalogo`. En `Main.java` crear objetos de tipo libro, insertarlos en el catálogo dado, y luego usar `buscarPorTitulo` para obtener libros del catálogo.

Ej. 3. Escribir una clase para el manejo de números racionales. Los atributos son dos variables de tipo long. Puede suponer que el denominador siempre es positivo. Escriba métodos sumar, restar, dividir, igual para las operaciones básicas sobre números racionales. Implemente un método toString para mostrar los números racionales por pantalla.

Ej. 4. Bajar los archivos `ej3a.zip` y `ej3b.zip` de Moodle, que contienen extensiones de las clases del ejercicio 1.

a) Compilar y ejecutar `Main.java` (de `ej3a.zip`). Qué diferencias observa entre las comparaciones de libros usando el operador `==` y el método `equals`? Implementar `equals` de `Catalogo.java`.

b) Compilar y ejecutar `Main.java` (de `ej3b.zip`). Teniendo en cuenta lo aprendido en el ejercicio anterior, explique a que se debe el error retornado (hint: el error se encuentra en la implementación de copy de `Catalogo`). Corrija la implementación de copy para evitar este error.

Ej. 5. Bajar el archivo `ej4a.zip` de Moodle. Este contiene las clases `Libro` y `Catalogo` del ejercicio 3, pero esta vez organizadas en paquetes. Compile estas clases utilizando directivas del compilador para generar los archivos `.class` en el directorio `bin` (separados del código fuente contenido en la carpeta `src`).

Ej. 6. Bajar el archivo `ej5.zip` de Moodle. Este contiene una interfaz `Lista`, una implementación genérica (que utiliza la clase `Object`) con listas enlazadas (clase `ListaSobreListasEnlazadas`), y algunos casos de prueba (clase `PruebaLista`).

a) Implementar los métodos faltantes de `ListaSobreListasEnlazadas`.

b) Es posible hacer una implementación genérica de listas en el lenguaje Pascal? (hint: ver ejercicios de la Práctica 1).

Ej. 7. Implemente el TAD pilas, para esto defina la interface de pilas con las operaciones usuales, y dé al menos dos implementaciones diferentes. Las implementaciones deben ser genéricas.

Ej. 8. Implemente el TAD colas, para esto defina la interface de colas con las operaciones usuales, y dé al menos dos implementaciones diferentes. Las implementaciones deben ser genéricas.

Ej. 9. Escribir métodos genericos en JAVA que calculen el mínimo y máximo de un arreglo.

Ej. 10. Escriba una clase que almacene una fecha, verificando que la misma sea correcta: esto es, que el día esté entre los límites 1 y 31 días del mes, que el mes esté entre 1 y 12 y que el año sea mayor o igual que 1582 (año gregoriano). Determine los atributos y métodos necesarios para el manejo de fechas, teniendo en cuenta que se deberá contar con un método bisiesto para verificar si la fecha que se quiere asignar es la correcta. Deberá controlar el acceso a los miembros de la clase, esto es determinar cuáles de ellos son privados, protegidos y/o públicos y justificar tu elección.

Ej. 11. Implemente en JAVA las colas de prioridad. Su implementación debe ser genérica utilizando la clase comparable.

Ej. 12. Escriba un programa que lea un texto y devuelva la cantidad de palabras con al menos tres vocales diferentes. Suponemos que cada palabra está separada de otra por uno o más espacios, tabuladores o saltos de líneas. La entrada de datos finalizará cuando se detecte la marca de fin de fichero.

Ej. 13. Implemente el TAD árboles en JAVA, un árbol consiste de una raíz, un hijo izquierdo y un hijo derecho (los cuales son árboles). Su implementación debe ser genérica.

Ej. 14. Busque información sobre la clase ArrayList, use esta clase para implementar una clase SortedArrayList; las instancias de esta clase contienen una secuencia de elementos ordenadas, y permiten las operaciones add, contains, get, isEmpty, remove y ensureCapacity.

Ej. 15.[Opcional] Se quiere realizar una clase para manejar laberintos, la clase tiene los siguientes atributos y métodos:

- Atributo `nombre`: el nombre del laberinto,
- Atributo `plano`: el plano del laberinto, este tiene que tener una entrada y una salida, paredes y pasillos, una forma de guardar el plano es con una matriz en donde se pueden representar los pasillos con 0's y las paredes con 1's.
- Atributo `posicion`: dice la posición actual en el laberinto,
- Método `crear`: este método crea un laberinto,
- Método `mostrar`: este método muestra un laberinto,
- Método `solucionar`: este método da una solución al laberinto, es decir, los pasos necesarios para resolver el juego.

Dar una implementación de esta clase en JAVA.